3. Decisions

Program Name: Decisions.java   Input File: decisions.dat

You are a strategic planner of routes. You need to deliver a bunch of goods to a relief zone from various warehouses. All of your warehouses have the appropriate goods, but the catch is, once you choose a warehouse, you cannot select any other warehouse to deliver the goods. As a result, you would like to make the trip from the warehouse to the relief zone as short as possible while only being able to travel up, down, left and right.

Input

The first line contains an integer, T, to denote the number of test cases. In each test case, the first line contains two integers, R and C, to denote the number of rows and columns in the map. For the following R lines, each line contains C characters for the map. Every character in the map will be one of the following characters:

- . (Period) – You can travel on this space
- # (Pound) - You cannot travel on this space
- W (Capital W) – These are the locations of your warehouses
- R (Capital R) – This is the relief zone you need to get to

Output

Please print out the row and column of the closest warehouse. If there is a tie, print out the row and column of the warehouse with the smallest row. If there is still a tie then, print out the row and column of the warehouse with the smallest column.

If there is no path to the relief zone, then print **Send a helicopter**.

Constraints

1 <= T <= 10
2 <= R <= 100
2 <= C <= 100
1 <= Number of warehouses <= 50

Example Input File

```
3
7 7
#W#.R##
...#.##
...#...
.##..W.
.#....W
.##.##.
.......
5 5
W.W.W
.....
W.R.W
.....
W.W.W
3 10
.W.###....
....###...
...###...R
```

Example Output to Screen

```
3 5
0 2
Send a helicopter
```